Transforms

⚠ DRAFT; Incomplete

On This Page

The Framework currently several transforms that can be used as part of a model transformation, as described below.

fluid.transforms.arrayToObject

Transforms an array into a JavaScript object.

back to top

fluid.transforms.arrayToSetMembership

back to top

fluid.transforms.arrayValue

Transforms a value into an aray. If the value is already an array, the expander will have no effect.

For example:

Start:

```
var source = {
    cat: "meow",
    sheep: [
        "baaa",
        "wooooool"
    ],
    ...
}
```

>>
Rule to convert to arrays:

```
var rules = {
    cat: {
        transform: {
            type: "fluid.transforms.arrayValue",
            inputPath "cat"
        }
    },
    sheep: {
        transform: {
            type: "fluid.transforms.arrayValue",
            inputPath "sheep"
        }
    }
    ....
}
```

>>
Result:

```
{
    cat: ["meow"],
    sheep: [
        "baaa",
        "wooooool"
    ],
    ...
}
```

Note that the value of `cat` is now an array, but the value of `sheep` is unaffected.

back to top

fluid.transforms.binaryOp

back to top

fluid.transforms.binaryLookup.*

back to top

fluid.transforms.condition

back to top

fluid.transforms.count

back to top

fluid.transforms.linearScale

back to top

fluid.transforms.literalValue

back to top

fluid.transforms.objectToArray

back to top

fluid.transforms.round

back to top

fluid.transforms.setMembershipToArray

back to top

fluid.transforms.value

This extracts and/or the value of a given path, and can be used for the following purposes:

To rename a property:

Start:

```
var source = {
    cat: "meow",
    ...
}
```

>>
Rule to rename "cat" to "feline":

```
var rules = {
    feline: {
        transform: {
            type: "fluid.transforms.value",
            // specify only the path to transform
            inputPath: "cat"
        }
    },
    ....
}
```

>>
Result:

```
{
    feline: "meow",
    ...
}
```

To set a default value:

Start:

```
var source = {
    gerbil: undefined,
    // or if "gerbil" doesn't exist
    ...
}
```

>>
Rule to set default value of "gerbil":

```
var rules = {
    gerbil: {
        transform: {
            type: "fluid.transforms.value",
            // specify path and default value
            inputPath "gerbil",
            value: "squeek"
        }
    },
    ....
}
```

>>
Result:

```
{
    gerbil: "squeek",
    ...
}
```

Note that if "gerbil" has a value initially, it will be unaffected.

To specify a literal value:

Start:

```
var source = {
    // no mention of kangaroos
    ...
}
```

>>
Rule to set a value for "kangaroo":

```
var rules = {
    kangaroo: {
        transform: {
            type: "fluid.transforms.value",
            // specify only a value
            value: "boingg"
        }
    },
    ....
}
```

>>
Result:

```
{
    kangaroo: "boingg",
    ...
}
```

To change the structure/nesting:

Start:

```
var source = {
    goat: false,
    sheep: [
        "baaa",
        "wooooool"
    ],
    ...
}
```

>>
Rule to change the nesting:

```
var rules = {
    "farm.goat": {
        transform: {
            type: "fluid.transforms.value",
            inputPath "goat"
        }
    },
    "farm.sheep": {
        transform: {
            type: "fluid.transforms.value",
            inputPath "sheep"
        }
    }
    ....
}
```

>>
Result:

```
{
    farm: {
        goat: false,
        sheep: [
            "baaa",
            "wooooool"
        ]
    },
    ...
}
```

back to top

fluid.transforms.value.invert

back to top

fluid.transforms.valueMapper

back to top